RL-TR-94-66
Final Technical Report
May 1994

NCTI SIMULATION MOD I

Synectics Corporation

Stephen J. Golembiowski

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

Rome Laboratory
Air Force Materiel Command
Griffiss Air Force Base, New York

This report has been reviewed by the Rome Laboratory Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RL-TR-94-66 has been reviewed and is approved for publication.

APPROVED: Alex

ALEX F. SISTI
Project Engineer

FOR THE COMMANDER: Garry W. Barringer

GARRY W. BARRINGER
Technical Director
Intelligence & Reconnaissance Directorate

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave Blank)

2. REPORT DATE
May 1994

3. REPORT TYPE AND DATES COVERED
Final Jul 92 - Sep 93

4. TITLE AND SUBTITLE
NCTI SIMULATION MOD I

5. FUNDING NUMBERS
C - F30602-92-C-0077
PE - 62702F
PR - 4594
TA - Q0
WU - 7N

6. AUTHOR(S)
Stephen J. Golembiowski

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Synectics Corporation
111 East Chestnut Street
Rome NY 13440

8. PERFORMING ORGANIZATION REPORT NUMBER
WH-92-OL-00

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Rome Laboratory (IRAE)
32 Hangar Road
Griffiss AFB NY 13441-4514

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
RL-TR-94-66

11. SUPPLEMENTARY NOTES
Rome Laboratory Project Engineer: Alex F. Sisti/IRAE/(315) 330-4518

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

The Non-Cooperative Target Identification (NCTI) project was created for Rome Laboratory (RL/IRA) to design and develop an enhanced sensor modeling capability in RL/IRA's Tac Brawler-based NCTI modeling environment. The objective of this effort was to develop an NCTI simulation and modeling test bed to provide an environment for development and verification of NCTI techniques. These techniques include the collection, fusion, dissemination, and utilization of advanced information. The test bed centers around a system to model air-to-air engagements. It is used to determine the changes in the engagement outcomes that would result from the availability of NCTI information. The resulting simulation allows the verification and quantification of the increased probability of kill (Pk) associated with various types of NCTI activities. The system allows the addition of future models and the substitution of existing models. It has the ability to integrate models that are co-resident, or that operate on a remote platform. The NCTI simulation and modeling test bed is supported by analysis and evaluation facilities to allow for a determination of the effects of NCTI info in an engagement.

14. SUBJECT TERMS
Non-Cooperative Target Identification, Model Management System, Software Backplane, Display System

15. NUMBER OF PAGES
48

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT
UNCLASSIFIED

18. SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

19. SECURITY CLASSIFICATION OF ABSTRACT
UNCLASSIFIED

20. LIMITATION OF ABSTRACT
UL

TABLE OF CONTENTS

1.0 INTRODUCTION ---------- 1
1.1 NCTI Development ---------- 1
1.2 NCTI Models ---------- 4
1.3 The NCTI Sim Mod I ---------- 4
1.3.1 The NCTI Simulation Mod I Backplane ---------- 4
1.3.1.1 Overview of System ---------- 4
1.3.1.2 Interface Libraries ---------- 6

2.0 SYSTEM FUNCTIONALITY ---------- 7
2.1 Supported Data Elements ---------- 7
2.1.1 Collection Models ---------- 8
2.1.1.1 Access to Ground Truth ---------- 8
2.1.1.2 Output Data ---------- 9
2.1.2 Fusion Models ---------- 9
2.1.2.1 External Interface Requirements ---------- 9
2.1.3 Dissemination Models ---------- 10
2.1.3.1 External Interface Requirements ---------- 10
2.1.4 Engagement Models ---------- 11
2.1.4.1 External Interface Requirements ---------- 11
2.2 Testbed Requirements ---------- 12
2.2.1 Interprocess Communications ---------- 12
2.2.2 Data Base Functions
2.2.2.1 Ground Truth Data Base ---------- 12
2.2.2.2 Perceived Truth Data Base ---------- 13
2.2.3 Synchronization ---------- 13
2.2.4 Evaluation ---------- 13
2.2.5 Control Functions ---------- 13
2.3 Model Management and Control ---------- 14

3.0 DISCUSSION OF THE BACKPLANE CONTROLLER ---------- 14
3.1 The Backplane Architecture ---------- 14
3.1.1 Data Management and Standardization ---------- 14
3.1.2 Interprocess Communications ---------- 17
3.1.3 Monitoring Functions ---------- 17
3.1.4 Display System ---------- 17

4.0 AREAS FOR FURTHER DEVELOPMENT ---------- 17
4.1 Data Dissemination Technologies ---------- 18
4.2 Correlation Types ---------- 18
4.2.1 Target-to-Target ---------- 19
4.2.2 Target-to-Track ---------- 19
4.2.3 Track-to-Track ---------- 19
4.3 Correlation Algorithms ---------- 19
4.3.1 Statistical Methods ---------- 21
4.3.2 Heuristic Techniques ---------- 21
4.4 NCTI Sensitivity Analysis ---------- 22
4.4.1 Confidence Levels ---------- 22
4.4.2 ID Levels ---------- 23
4.4.3 Effects of Temporal Distortion ---------- 23
4.4.4 Impact Analysis for Weapons Deployment ---------- 23
4.4.5 Tracking Error Analysis ---------- 23

TABLE OF CONTENTS (cont'd)

5.0 BACKPLANE UTILIZATION --- 23
- 5.1 Use of Model Integration Library Resident in Mod I --- 24
- 5.2 Data Requirements and Units Conversion --- 24
  - 5.2.1 Spatial Conversion Routines --- 24
  - 5.2.2 Kinematics --- 25
  - 5.2.3 Radio Frequency (RF) Emissions --- 25
- 5.3 Communication with Backplane --- 26
- 5.4 Model Startup, Initialization, Synchronization --- 26
- 5.5 Integration of a Model --- 27
  - 5.5.1 Input Data Requirements Analysis and Design --- 28
    - 5.5.1.1 Data Contents --- 28
    - 5.5.1.2 Data Representation --- 28
    - 5.5.1.3 Data Access --- 29
  - 5.5.2 Output Analysis and Design --- 31
    - 5.5.2.1 Sensor Model Analysis and Design --- 31
    - 5.5.2.2 Report Dissemination Strategies --- 31
  - 5.5.3 Timing Analysis and Design --- 31
    - 5.5.3.1 Minor vs. Major Ticks --- 31
    - 5.5.3.2 Normal vs. Fast Ticks --- 33
    - 5.5.3.3 Event-Driven and Monte Carlo Models --- 33
    - 5.5.3.4 Deterministic Models --- 33
- 5.6 Straw Man Model Code Sample --- 35

List of Exhibits

1 NCTI Concept --- 2
2 Concept of Correlation and Fusion of Sensor Data --- 3
3 NCTI Modeling Environment Subsystems --- 5
4 NCTI Model Management System --- 15
5 NCTI Testbed Architecture --- 16
6 Basic Correlation Types --- 20
7 Techniques for Sensor Data Interfacing --- 30
8 Template for Mod I Function --- 32
9 Integration of a Sensor Model into the Backplane Environment --- 34

1.0 INTRODUCTION

Non-Cooperative Target Identification (NCTI) refers to the long range acquisition and identification of hostile aircraft. It further includes the determination of as much characteristic information as can be derived concerning those aircraft. Some of these characteristics include:

1. Identification of airframe type to determine maneuverability, weapons potential, flight radius, and potential roles in an engagement.

2. Determination as to stores on board and their potential.

3. Command and control frequencies and structures being utilized by opposing ground and air elements.

The ultimate goal of such identification is to produce accurate and highly detailed models of an opposing air threat, and to provide pertinent information to friendly assets to increase kill ratios.

As might be expected the construction of an infrastructure to collect, process, and disseminate information of this type is an incredibly expensive and technically challenging one. In addition (as with any highly complex system) the actual behavior and benefits to be gained from such a system are difficult to predict. Exhibit 1 illustrates one possible concept for an NCTI construct. The complexities and variability in such a system are immediately evident.

1.1 NCTI DEVELOPMENT

An understanding of the history and rationale surrounding NCTI, NCTI modeling, and NCTI processing helps to clarify the need for the NCTI Sim Mod I effort. The Air Force has had an officially recognized need for NCTI dating back to the 1960s. In those years, NCTI was primarily directed at the shooter platform. This need was based on the availability of air-to-air missiles that could be launched and guided to targets at ranges equivalent to the radar lock-on range, and on rules of engagement (ROE) which demanded positive identification of targets for weapon launch. At that time this identification could only be accomplished by a wingman proceeding to the target for a visual ID, which was then transmitted to the shooter platform for weapon release—a highly dangerous and inefficient process.

In the mid 1960s the Air Force, in concert with the Navy, launched a serious developmental effort for NCTI techniques. These contractual efforts were performed under a program at Wright Laboratory called PAVE GAMMA. This and other early systems were limited by the technology available for their implementation.

As these NCTI technologies matured the Air Force realized in the early 1970s that a strong need existed and had to be ratified to "fuse" all of the burgeoning number of NCTI data sources. The goal of this requirement was to achieve accurate correlation and association of information from multiple sources to generate an overall "picture" of a battle or theater. Exhibit 2 is an example of the notion of correlation for C3I and NCTI.

Many fusion processes have been proposed, investigated, and developed, under Department of Defense (DoD) funding. To date though, neither an airborne long range surveillance

Exhibit 1
NCTI Concept

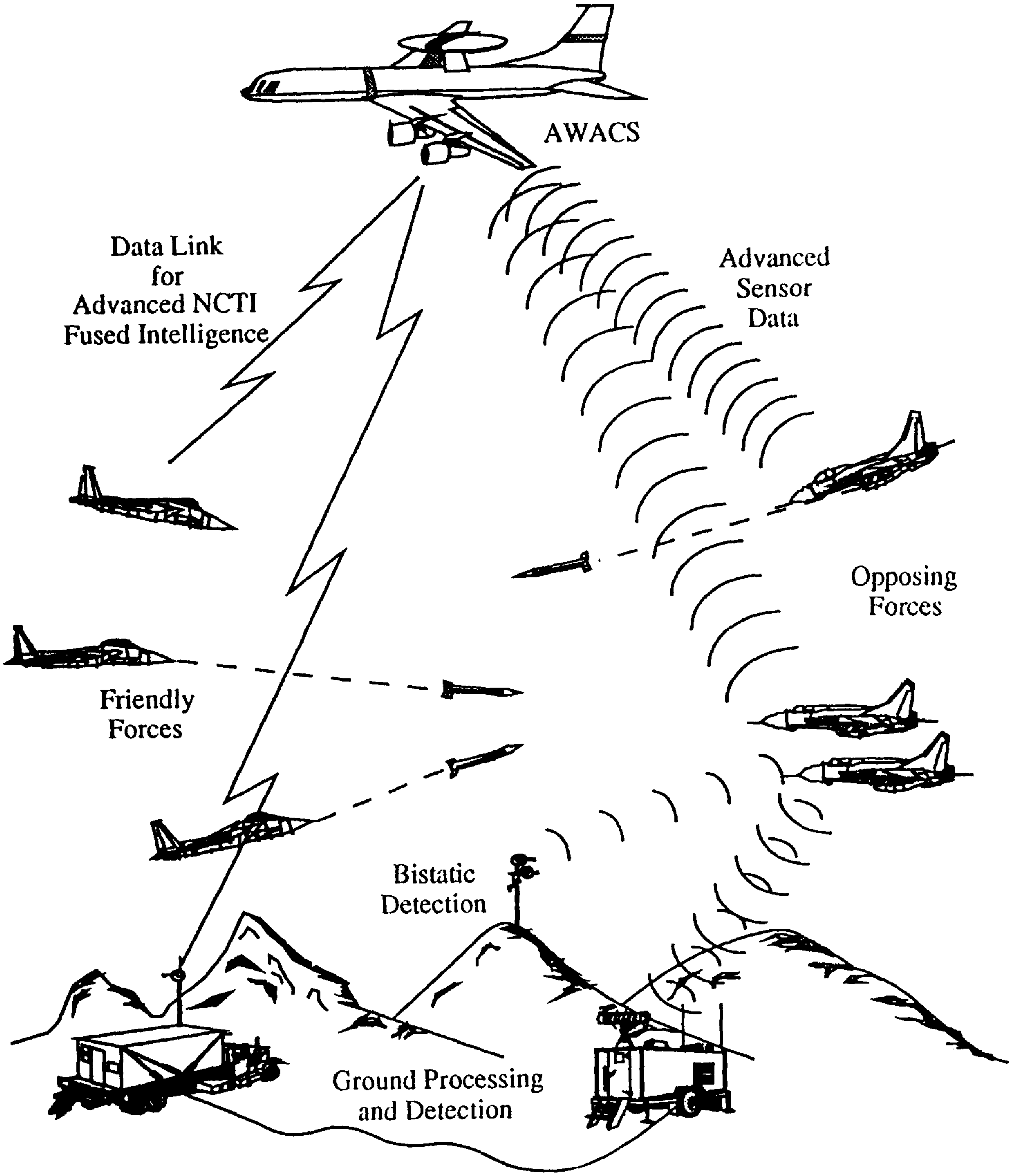
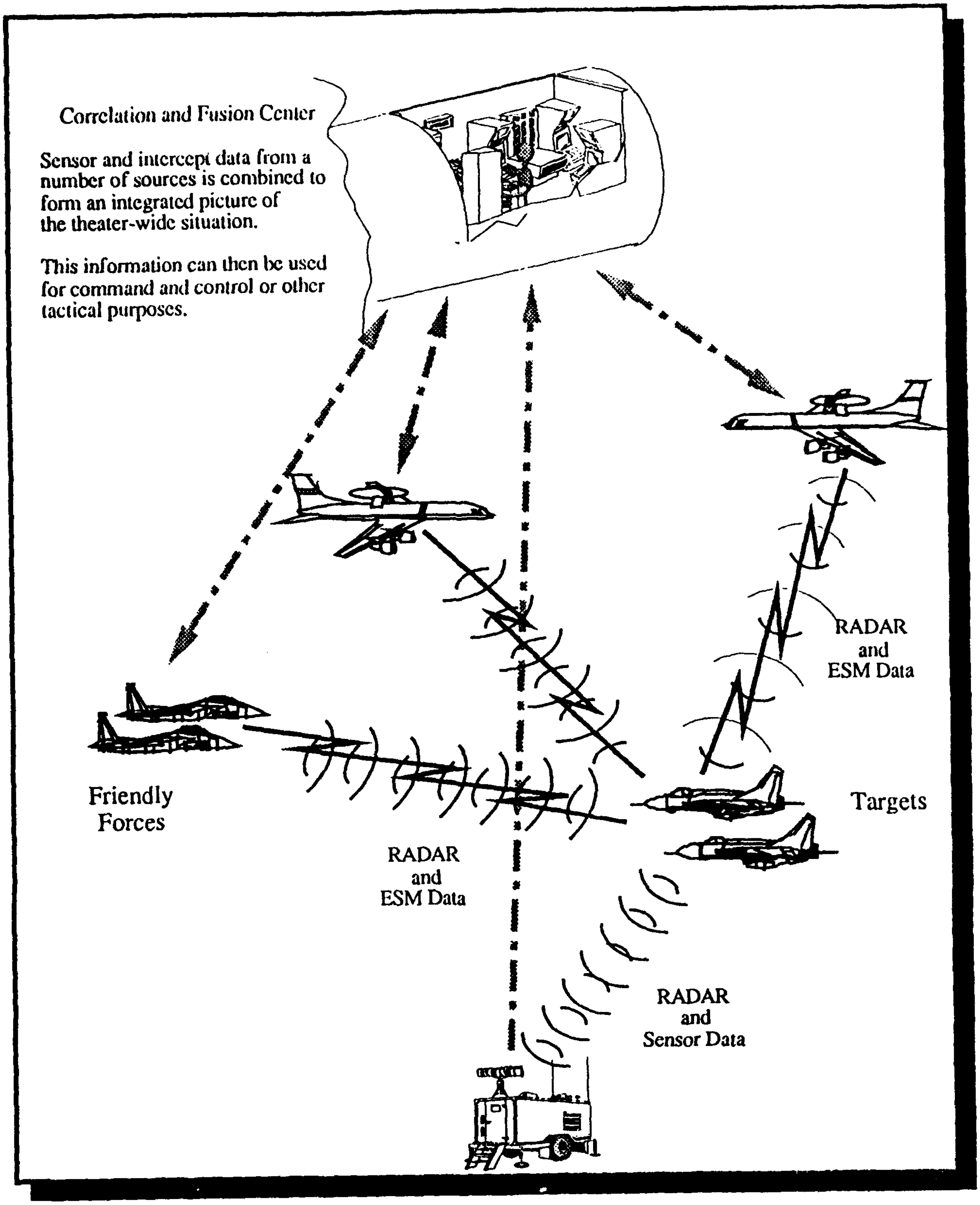


Exhibit 2
Concept of Correlation and Fusion
of Sensor Data


Correlation and Fusion Center
Sensor and intercept data from a number of sources is combined to form an integrated picture of the theater-wide situation.
This information can then be used for command and control or other tactical purposes.
RADAR and ESM Data
Friendly Forces
RADAR and ESM Data
Targets
RADAR and Sensor Data

platform (e.g., E-3) nor an airborne shooter platform (e.g., F-15) has an operational, fully automated fusion process for identification (although many laboratory systems are currently, or soon will be, in operation at Rome Laboratory and other locations).

Over the years, the Air Force has continued to maintain an official requirement for NCTI and has pursued this requirement via various required operational capabilities (ROC), statement of need (SON), etc., and has awarded many contractual efforts to pursue various NCTI technologies via Wright Laboratory, Rome Laboratory, and Electronic System Center. The Navy has funded NCTI efforts via the Naval Research Lab (NRL), NAWC, NWC (China Lake), and NOSC (San Diego).

1.2 NCTI MODELS

The net result of all of these diverse activities is that a large number of NCTI relevant techniques and algorithms have been developed independently by a large group of researchers. The overall goal of this NCTI modeling and simulation effort was to allow these diverse elements to be combined so that researchers could model the overall NCTI environment, including the collection, fusion, dissemination, and utilization of advanced information, and could determine the changes in the engagement outcomes that would result from different configurations and constructs. Modeling of this nature allowed determinations to be made as to the effectiveness of the NCTI data as collected and passed to the pilots. Probabilities of kill (Pk) and fratricide (Pf) and a host of other costs and benefits could be evaluated. Realistic engagement models run both with, and without, the additional information provided by the NCTI suite, give details of effectiveness, and point up areas for further investigation.

1.3 THE NCTI SIM MOD I

The NCTI Sim Mod I testbed system is an environment which allows the integration, in a cohesive fashion, of a number of divergent models and components. The purpose of these elements is to simulate and evaluate the NCTI tactical environment, and the effect of NCTI information as a force multiplier.

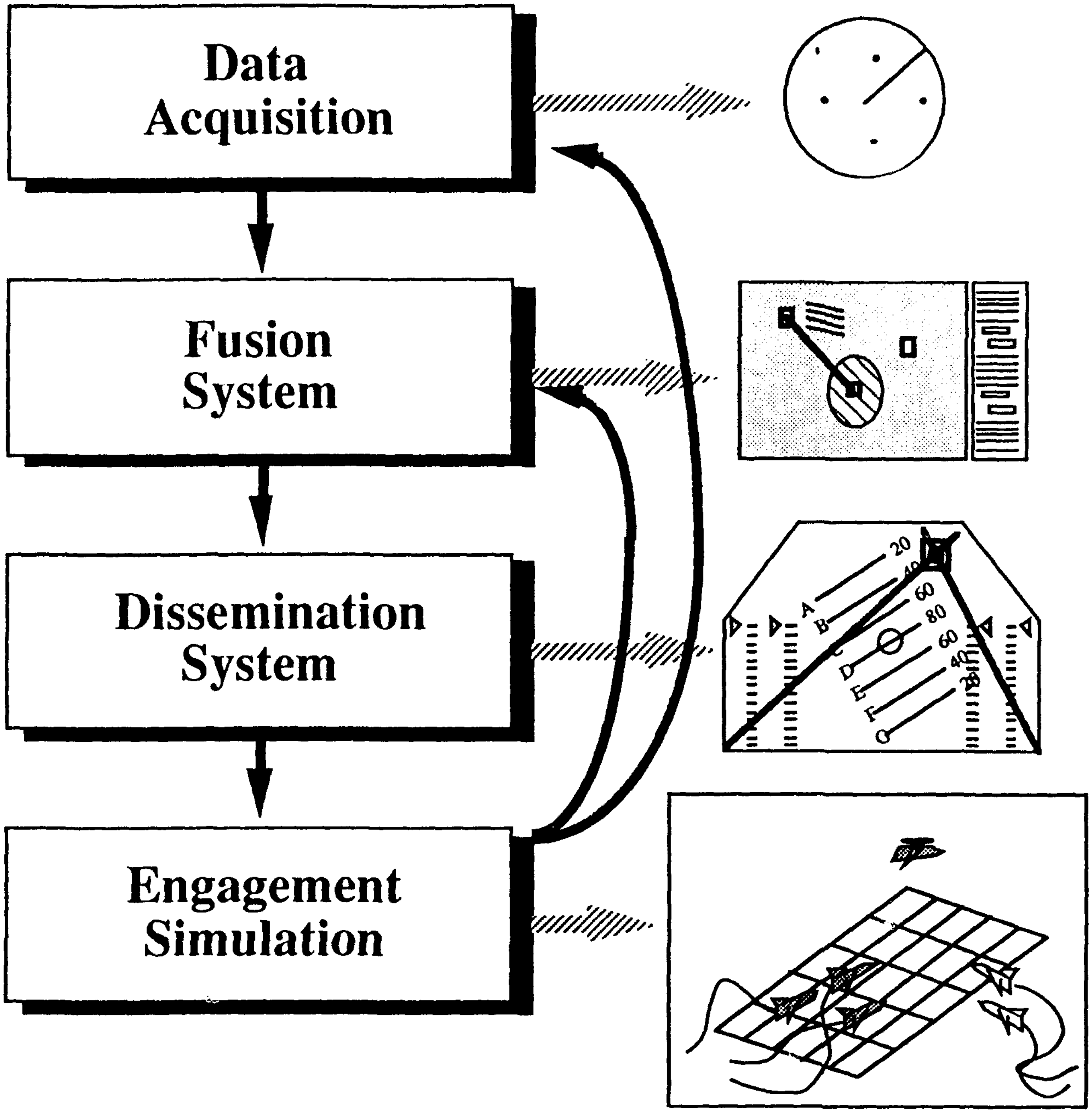
1.3.1 THE NCTI SIMULATION MOD I BACKPLANE

The NCTI Simulation (Sim) Mod I Backplane is an environment (illustrated in Exhibit 3) that provides management and support for the four major NCTI modeling component types. These are data acquisition systems (sensor models), data fusion and integration system (fusion), data dissemination, and the combat model (engagement model) such as the TAC Brawler. An understanding of the Backplane, its architecture, and its utilization is crucial to the success of this effort. We present our understanding in the remainder of this section.

1.3.1.1 Overview of System

The Backplane is implemented in two general pieces. The first component is the Backplane controller which serves as a resource manager, interprocess communication facility, and data base manager for the representative data bases. The second component is a library of functions which

Exhibit 3
NCTI Modeling Environment Subsystems



allow individual models of each of the four major types (collection, fusion, dissemination/ controller, and engagement) to access the resources of the Backplane controller.

The software requirements of the Backplane process itself are fairly modest compared with the processes that were to be constructed around it. Software is required to manage data, to pass messages between models, to control access to the data bases and other resources, and to provide timing and synchronization for models. Additional software is used to monitor and control the Backplane, to gather statistics, and to display information about models.

The data manager is required to manage two primary data base types. The first is the data base which represents the ground truth; the second is the data base that represents the perceived world as assembled by the collectors and fusion processors.

The message handler is used to pass target reports from the collectors to the correlators, and commands and tactical data from the disseminators to the combatants. Additional side channels are required to return replies to the disseminators / controller from the combatants.

The resource manager is used to control access to the data base and other resources. The NCTI testbed is predicated on the notion of multistep access to data elements. To ensure the integrity of the data between steps, mutually exclusive access is granted to various processes via the resource manager. This is the equivalent of a simple, full possession, lock manager. Although more efficient access schemes are available, the simplicity of integration into existing models represented by this scheme makes it superior for the NCTI Sim Mod I testbed.

The timing system keeps track of the asynchronous processes (models) that make up the testbed as a whole, and ensures that all of the elements are processing in the same time domain (scenario second) as the run progresses.

The Backplane includes a set of control and display functions which provide the ability to view the data bases; statistics concerning the data traffic; statistics concerning functionality of various models; and evaluation tools.

1.3.1.2 Interface Libraries

The second set of functionality involves the library routines which allow models, both new and existing, to access the Backplane system. These are divided into collection, fusion, dissemination / controller, and engagement. A subset of the functionality provided includes:

1.3.1.2.1 Interprocess communications. Interprocess communications between the models are performed via the Backplane and are standardized to allow for the system to operate correctly and efficiently. By using the UNIX socket mechanism, processes are able to communicate with each other through a common protocol and thus support is available for a wide range of hosts and operating systems.

1.3.1.2.2 Data base access. A library of functions facilitates all access to the ground truth and perceived truth data bases. The library supports the following operations to manipulate records: add, delete, retrieve, search, scan, update, request, lock, release, and declare.

1.3.1.2.3 Synchronization. The Backplane system provides a mechanism to synchronize each of the individual model processes. Because processes depend on others for various kinds of information in a time dependent manner, each component is required to access the system time for proper communication and synchronization with each other.

1.3.1.2.4 Evaluation. The Backplane includes an evaluation module which will provide the system accuracy information about the fusion processes. This module examines the records in the ground truth data base (ground truth IDs) and compares them to the records in the perceived truth data base, and generates statistics as to the accuracy of the fused data. This allows for a standardized method of evaluating a number of different algorithms. Similar evaluation functions exist to determine Pk and other NCTI specific information.

1.3.1.2.5 MOTIF GUI support. By using a pictorial metaphor for all of the objects and operations within the NCTI processing environment, the Backplane system allows the user to concentrate more energy on the tasks being performed and less on the mechanics of performing it. Visually presented information can be accessed by human perception in a most natural way. Complex structures and relations can be perceived in less time, in greater number, and with less errors than in any other way. Models of the real world or models of abstract concepts are hardly dealt with by users without resorting to visual representations.

MOTIF is the graphics windowing package of choice because of its portability and widespread support throughout the hardware and software community. It is becoming the standard among windows packages for workstations throughout the industry.

2.0 SYSTEM FUNCTIONALITY

This section specifies the functionality provided by the NCTI Sim Mod I Backplane controller and by the series of library interface routines which comprise the core of an NCTI Backplane architecture. In addition it spells out specifics of the software requirements for each of the four model types identified in the Functional Requirements Document—collection model, fusion model, dissemination model, and engagement model—in order to utilize that architecture.

2.1 SUPPORTED DATA ELEMENTS

NCTI modeling requires sufficient data to allow the associated models to make the determination of as much characteristic information as can be derived concerning modeled aircraft. All of these characteristics can be summed up to form three broad categories of NCTI information:

1. Identification of airframe type to determine maneuverability, weapons potential, flight radius, and potential roles in an engagement.

2. Determination as to stores on board and their potential.

3. Command and control frequencies and structures being utilized by opposing ground and air elements.

The ultimate goal of such identification is to produce accurate and highly detailed models of an opposing air threat, and to provide pertinent information to friendly assets in order to increase kill ratios.

This data is used in many forms by the four major NCTI modeling components. By way of review these components are data acquisition system (collection model), the data fusion and integration system (fusion model), the data dissemination system (dissemination model), and the combat model (engagement model). Each of these components and their data requirements will be discussed in the following sections.

2.1.1 COLLECTION MODELS

The collection portion of the modeling environment simulates the actual detection and tracking assets that can be brought to bear against hostile targets. It is at this level that such operations as voice and background recognition, high definition radar imaging, emission analysis, and ingress-egress intelligence are collected and preliminarily processed.

2.1.1.1 Access to Ground Truth

The collection models examined parametric data on aircraft, emitters, and other elements of interest which are stored in the ground truth data base. Access to this data base is through a specialized set of access routines. These access routines must be used to read and write data appropriately. Data read will consist of the following.

2.1.1.1.1 Aircraft parameters. Aircraft parameters required for the collection models include, but are not limited to:

√ Location (x,y,z) within the modeled space
√ Emitter states
√ Engine status and throttle position
√ Stores on board
√ Tail number
√ Aircraft type
√ Aircraft length
√ Speed
√ Fuel capacity
√ Maximum operating range

2.1.1.1.2 Sensor platform parameters. Sensor platform parameters required for the collection models include, but are not limited to:

√ General class
√ Effective ranges
√ Antenna gain
√ Operating frequency range
√ Frequency modes
√ Modulation type
√ Traffic type
√ Pulse repetition interval
√ Range of possible pulse durations
√ Bandwidth
√ Power level
√ Orientation
√ Field of view

Other parameters should include, but are not limited to: terrain elevation, weather, probability of detection, area of coverage parameters, and sensitivity parameters.

2.1.1.1.3 Control parameters. After the sensor model has completed its calculations, the software takes over to allow the results to be fed into the fusion model. Control parameters such as timestamp information allow the synchronization of the entire model environment. The sensor output and control parameters should include at least the following: parameters for the insertion of external models, current simulation time, and flags to control access to various resources.

2.1.1.2 Output Data

Output data consists of individual target reports which contain all collectible parametric data for an airframe or ground target, or an emitter. The form of this information must be standardized in NCTI testbed form and must be queued for use by the fusion algorithms via the Backplane message processor.

2.1.2 FUSION MODELS

The fusion system orders, fuses, and augments the raw data collected in the collection system. It is at this level that characteristics for individual airframes are amassed, and that those airframes are tracked. It is at this level, for example, that a background noise analysis that determines engine type, and a report of a particular aircraft's earlier takeoff, would be correlated to determine possible mission postures.

2.1.2.1 External Interface Requirements

The data and interface requirements for the fusion models consist primarily of the sensor data inputs, and of the perceived (fused) data base as output. There is also the requirement for timing and control information to synchronize any fusion model(s) which might be installed. The data and interface requirements for the fusion model are discussed in the following sections.

2.1.2.1.1 Sensor data in input. The first input channel includes the information that is output by the sensor model(s) and includes at least the following:

√ *Position of target within the space.* Many sensors will include velocity and heading information in these data.

√ *Sensor specific information.* The data provided by a sensor are subject to wide variance based on the sensor type modeled.

√ *Tail number or ground truth ID.* These elements are passed through from the ground truth and are to be used for evaluation purposes only.

2.1.2.1.2 Perceived world (fused data base) data out. All of the parameters concerning the perceived disposition of all of the aircraft, platforms, emitters, and other tactically significant elements in the model environment are output into the fused data base by the fusion models. These data elements include at least the following:

√ *Element position within the modeled space*, including speed, heading and altitude, and current state of emitters (e.g., operational capacity of radar and radio; frequencies being used, etc.).

√ *NCTI derived information* such as engine status, fuel load and stores on board.

√ *Tail number or ground truth ID.* These elements are passed through from the ground truth and are to be used for evaluation purposes only.

2.1.2.1.3 Control parameters. There must be a mechanism whereby the fusion model(s) can be synchronized with other elements of the modeling environment. Control parameters include timestamp information to allow the synchronization of the entire model environment, and semaphores for requesting access to data and channels, and for signaling the completion of local additions and updates to data elements. These mechanisms are similar to those found within the collection and dissemination models and include:

√ *Data stream to and from the sensor models*. Libraries of routines to allow both internal processes and external processors to access the channels discussed earlier must be provided.

√ *Tick*. The tick parameter set must include the current simulation time as well as a flag which can act as an interrupt for each simulation second that passes.

2.1.3 DISSEMINATION MODELS

The dissemination model is a broad model which encompasses a number of smaller elements. Within this model such things as data interpretation by analysts and controllers, data and command links, tactical display types, data loads, and assimilation capabilities of controllers, are simulated.

2.1.3.1 External Interface Requirements

The data and interface requirements for both the human operator and automated models of data dissemination are essentially the same. These are discussed in the next few sections.

2.1.3.1.1 Perceived world (fused data base) data in. All of the parameters concerning the perceived disposition of all of the aircraft, platforms, emitters, and other tactically significant elements in the model must be available to the dissemination models. These data elements are used in the decision logic employed by the command models, pilot models, and other elements involved in the engagement model. These data elements include at least the following:

√ *Element position within the modeled space*, including speed, heading and altitude, and current state of emitters (e.g., operational capacity of radar and radio; frequencies being used, etc.).

√ *NCTI derived information* such as engine status, fuel load and stores onboard.

√ *Tail number*. As with the other testbed components, it is required that the tail number (ground truth ID) be available for this model so that it can perform its internal housekeeping operations. This is not to say that the controller would have access to enemy tail numbers in a real environment.

2.1.3.1.2 Command out channel. The second channel includes the information that is transmitted (disseminated) to the pilots involved in the engagement and will include at least the following:

√ *Vectoring (or waypoint) information and command data of the traditional (existing) type*. This includes voice and some digital data. IFF and other traditional information would also be included in this category.

√ *NCTI tactical data*. This includes the advanced data, sent via secure link, to be displayed in the cockpit, or to be utilized by onboard processing assets.

2.1.3.1.3 Reply back channel. The third dissemination interface channel includes the information that is transmitted by the pilots involved in the engagement to the disseminator controller and includes at least the following:

√ *ACK, NAK (acknowledged or actively not acknowledged.* (Note: This is different from NRPLY, no reply). This includes voice and some digital data. IFF and other transponder or beacon information would be included in this category.

√ *Uplink data.* In models where the sensor information and collection assets of the aircraft involved in the engagement are to be used in fusion operations and disseminated to other combatants (Situation Awareness Networks or SANs), this channel is active. Note that this channel is also available to the fusion model(s).

2.1.3.1.4 Control parameters. As the dissemination model performs its calculations there must be a mechanism whereby the results can be synchronized with other elements of the model. Control parameters include timestamp information to allow the synchronization of the entire model environment; and semaphores for requesting access to data and channels, and for signaling the completion of local additions and updates to data elements. These mechanisms are similar to those found on the collection and fusion models and include:

√ *Data stream to and from the dissemination subsystem.* Libraries of routines to allow both internal and external processes to access the channels discussed earlier must be provided. These external processes may be either resident on the same host or on an external processor.

√ *Tick.* The tick parameter set must include the current simulation time as well as a flag which can act as an interrupt for each simulation second that passes. The tick parameter is necessary to coordinate the Monte Carlo nature of some event-driven models with the time oriented data requirements of deterministic models.

2.1.4 ENGAGEMENT MODELS

The engagement model is the system which evaluates the interaction between friendly and hostile assets. This is the point at which the effectiveness of the NCTI data as collected, fused, and passed to the pilots can be evaluated. Realistic engagement models run both with, and without, the additional information provided by the NCTI suite, give details of effectiveness, and point up areas for further investigation.

2.1.4.1 External Interface Requirements

The information supplied by the engagement model to be used eventually by sensor models should be at least the physical state data. This includes vector position, velocity, and body orientation information. Additional information supplied should include acceleration, body angular rates, and engine state information. Certain identification data is needed to appropriately label each target. This includes target type information and a unique identifier. The engagement model should also supply RF emissions data for ESM sensor modeling. This includes information on RF emissions during the preceding frame, radar, jammers, and voice communications.

Track information such as position, velocity, and state time should be sent to the engagement model. This includes any one of the following:

√ Cartesian x, y, z, vx, vy, vz in a flat earth coordinate system.

√ Latitude, longitude, altitude, speed, heading, rate-of-climb.

√ Range, azimuth, elevation, range rate, azimuth rate, and elevation rate, relative to a specified observer location.

The physical information should also have confidence data associated with it.

2.2 TESTBED REQUIREMENTS

The NCTI testbed is a complex system composed of many different parts. Discus below are some basic functions which are required for the system to operate correctly efficiently.

2.2.1 INTERPROCESS COMMUNICATIONS

Interprocess communications must be standardized to allow for the system to operate correctly and efficiently. By using the UNIX socket mechanism, processes are able to communicate with each other through a common protocol.

2.2.2 DATA BASE FUNCTIONS

Several functions are needed to access the ground truth and perceived truth data bases. These are detailed in the next two sections.

2.2.2.1 Ground Truth Data Base

This library allows a wide variety of operations to be performed on the ground truth data base. Software allows performance of the following operations:

√ ADD – Add the contents of a data record to the data base.

√ DELETE – Delete the contents of the indicated data record from the data base.

√ RETRIEVE – Retrieve and transmit to the requester the contents of the specified data record.

√ SEARCH – Search the data base using specific parameters and transmit to the requester all records that match the criteria.

√ SCAN – Return the entire contents of the data base to the requester.

√ UPDATE – Replace the contents of the specified data record with a new data record.

√ REQUEST – Request that other processes (i.e. specified ones) not access the data base.

√ LOCK – Lock the entire data base so that only the requester can utilize the data base.

√ RELEASE – Terminate a previous REQUEST or LOCK.

2.2.2.2 Perceived Truth Data Base

This library allows a wide variety of operations to be performed on the perceived truth data base. Software allows performance of the following operations:

√ ADD – Add the contents of a data record to the data base.

√ DELETE – Delete the contents of the indicated data record from the data base.

√ RETRIEVE – Retrieve and transmit to the requester the contents of the specified data record.

√ SEARCH – Search the data base using specific parameters and transmit to the requester all records that match the criteria.

√ SCAN – Return the entire contents of the data base to the requester.

√ UPDATE – Replace the contents of the specified data record with a new data record.

√ REQUEST – Request that other processes (i.e. specified ones) not access the data base.

√ LOCK – Lock the entire data base so that only the requester can utilize the data base.

√ RELEASE – Terminate a previous REQUEST or LOCK.

2.2.3 SYNCHRONIZATION

The Backplane system has a mechanism to synchronize each of the individual processes. Because processes depend on others for various kinds of information in a time dependent manner, each component is required to access the system time for proper communication and synchronization with each other.

2.2.4 EVALUATION

The NCTI Simulation software contains an evaluation module which provides the system accuracy information about the fusion process. This module examines the records in the ground truth data base (ground truth IDs) and compares them to the records in the perceived truth data base and generates statistics as to the accuracy of the fused data.

2.2.5 CONTROL FUNCTIONS

Each component of the system has the requirement to, on demand, toggle inputs and/or outputs to each process. Each process has associated with it a window user interface to accomplish this and other various tasks. One of the other tasks is to generate statistics about each process' CPU time, memory allocation, percent of the total system time, etc.

2.3 MODEL MANAGEMENT AND CONTROL

In order to use the NCTI Sim Mod I, a system for controlling and configuring the models and the Backplane is required. The NCTI Sim Mod I system includes the Model Management System to perform this task.

The NCTI Model Management System is essentially a configuration management system that generates input to the Backplane process. As Exhibit 4 shows, the selection boxes on the left-hand side of the screen allow the user to build his/her system that is displayed graphically on the right-hand side of the screen. For instance, in this example there are two sensor models (Sensor1 and Sensor2) whose output is sent to Queue1. From that point, data is fed into a fusion model (Fusion1) where a perceived truth data base is built called Database. The dissemination model (Dissem1) gets it data from Database and feeds it through to the engagement model (TAC Brawler).

3.0 DISCUSSION OF THE BACKPLANE CONTROLLER

This section describes the high level architecture implemented in the NCTI Sim Mod I system to support the functionality described in Section 2 of this document, and in the Functional Requirements Description.

3.1 THE BACKPLANE ARCHITECTURE

The divergent nature of the processes and models that make up the NCTI testbed made it imperative that a central process be constructed to integrate all of the elements. This central process, known as the Backplane, serves as a standardized, flexible framework on which to hang individual models. Exhibit 5 illustrates the resulting architecture.

The Backplane structure is required to provide the following functionality:

3.1.1 DATA MANAGEMENT AND STANDARDIZATION

The Backplane supports the least common denominator of data needed to achieve meaningful representation of both the ground truth (real) and the perceived truth (as viewed through sensors). In addition, the Backplane implements a straightforward, standardized technique for accessing this information, and has controls to guarantee the integrity of these data as they are being accessed by multiple, asynchronous processes.

Exhibit 4
NCTI Model Management System

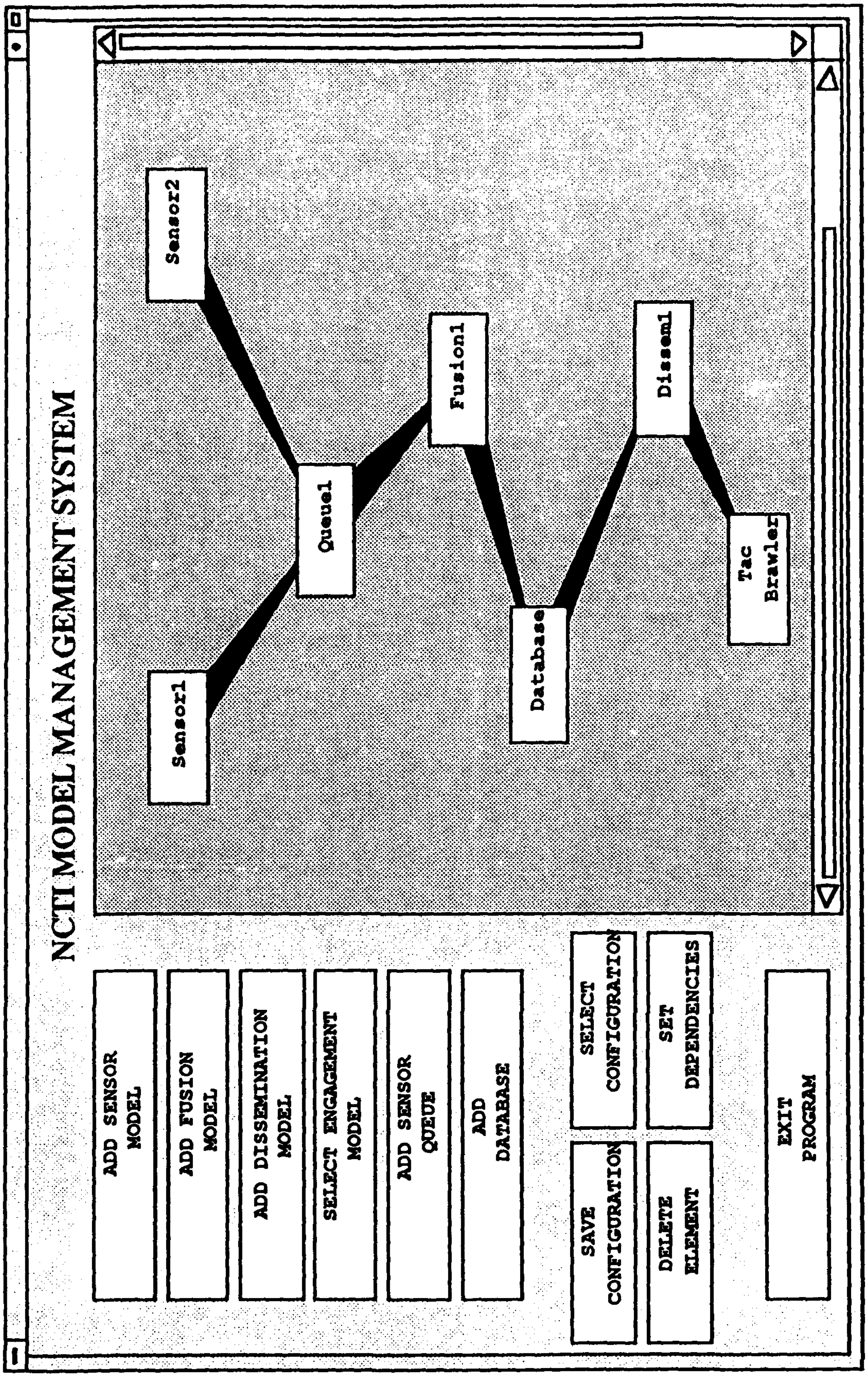
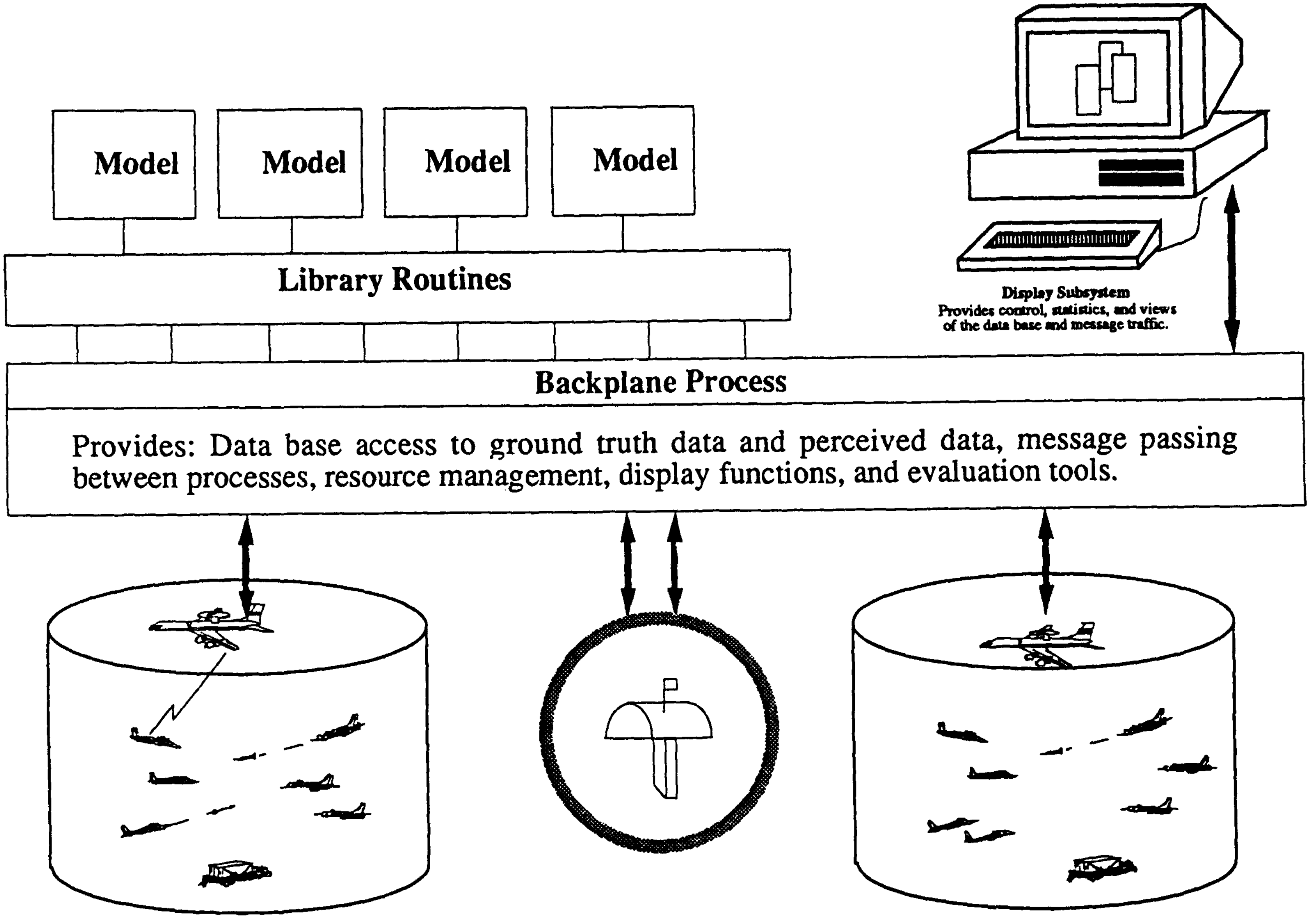


Exhibit 5
NCTI Testbed Architecture



Provides: Data base access to ground truth data and perceived data, message passing between processes, resource management, display functions, and evaluation tools.

Display Subsystem
Provides control, statistics, and views of the data base and message traffic.

Ground Truth
Data base contains positions and activity parameters for all objects in the model.

Message Handler
Ring buffers used to route messages and broadcasts to other recipient processes. Messages are standardized and monitored by Backplane.

Fused Data Base
Data base contains positions and activity parameters for all objects in the model *as perceived by the sensor models and fused by the fusion models.*

3.1.2 INTERPROCESS COMMUNICATIONS

The Backplane provides the functionality for interprocess communications among the various models. To allow the processes to communicate freely among themselves without going through the Backplane was to invite endless integration problems as the testbed evolved. Also, such standardization allowed for the construction of tools for the monitoring and control of such communication.

3.1.3 MONITORING FUNCTIONS

If all relevant data, and all communications and commands are passed through a common point, and in a common format, the ability to monitor and control these communications becomes inherent. This ability is especially important in the context of an NCTI testbed since it provides valuable insight into a number of parameters which might otherwise be difficult to collect.

Some of these include:

√ Communications densities – how much data is passed to the combatants; how many target reports are made to the fusion elements; etc.

√ Correlation accuracy – since the ground truth and perceived truth are stored in a known format, and since all updates are performed by Backplane processes, very meaningful statistics concerning accuracy and failure modes of correlation and fusion may be developed.

3.1.4 DISPLAY SYSTEM

The final area where a Backplane system shines is in the area of display generation. A large part of constructing any model is the generation of systems to display the results. Since the data storage is standardized through the Backplane, NCTI models are relieved of a good deal of this burden. Rather the Backplane processor implements tools to view data, messages, and so on. This results in great efficiencies of implementation for model builders.

4.0 AREAS FOR FURTHER DEVELOPMENT

The Air Force, and Rome Laboratory, have invested a great deal of effort in the development of modeling and simulation systems for the NCTI arena. With the completion of the NCTI Sim Mod I effort the mechanism is in place to permit the exploitation of the system's models which have been developed so far.

The NCTI Sim Mod II effort will result in a significant library of models which may be integrated, through the Mod I Backplane, into a single conceptual modeling environment. This will represent a significant step forward in the ability to model complete, end-to-end, NCTI environments.

The crucial step now is to take advantage of this modeling capability in order to perform tests which may be of significance to those constructing NCTI systems.

There are a number of NCTI relevant issues that we feel are critical in understanding NCTI and NCTI modeling to the extent required for the successful realization of the original goals of NCTI. These topics include understanding the data links used to transmit NCTI data, and an understanding of the correlation techniques applicable to NCTI, among others. The remainder of this section suggests some of the types of experimentation that would be pertinent to NCTI development and to the development of understanding in these areas.

4.1 DATA DISSEMINATION TECHNOLOGIES

It is important to examine the data links involved in processing NCTI data. This examination includes friendly force data links (as a possible limitation) and hostile force data links (as a potentially exploitable resource). The friendly forces' data links can be divided into two main functions. First, there is the adequacy, or lack thereof, of data links providing information from potential NCTI data sources off board to an airborne C3I platform (e.g., E-3). These offboard sources include JOINT STARS, SIGINT platforms (e.g., RIVET JOINT), GTACS sites, and National and Theater INTEL sources. Second, the data links to the shooter platform provide long range NCTI information and at some point can be used with the shooter's own onboard NCTI sources. Currently, the E-3A has only voice communications with the shooter platforms. Air Force efforts are ongoing to improve the performance of all the above data links in terms of speed, information capacity, commonality, and resistance to threat ECM.

SIGINT platforms that monitor threat communication channels and attempt to identify language and speaker ID and to merge content, currently perform these processes *manually*. There are several Air Force efforts, sponsored by Rome Laboratory, which are developing algorithms to perform the above processes automatically. In addition, there are algorithms being developed that process cockpit background from which airframe ID is accomplished.

4.2 CORRELATION TYPES

The design and implementation of an NCTI modeling system requires a thorough understanding of correlation and fusion algorithm types, their applicability to various data and sensor types, and their suitability based on target and data density, computational availability, and data storage and retrieval rates for the target platform. Additionally, a thorough understanding of the concepts surrounding track generation and maintenance is required and an understanding of identification techniques and paradigms is needed.

Correlation is the merging of diverse data into a single coherent representation of the perceived "Tactical Situation." As a practical matter, no single sensor or intelligence source can satisfy all of the information needs of C3/NCTI controllers and combatants. Each of today's sensors has inherent limitations; for example certain sensors can collect only against enemy equipment that is currently emitting RF radiation.

Additionally, a single source of information would leave consumers vulnerable to the loss of that source. Therefore, multiple sources and methods are required to collect the information necessary for understanding of the tactical situation. Correlation forms the basis for analyzing, evaluating, aggregating, and drawing conclusions from masses of imperfect or inconclusive data and merging the results into a single coherent representation of the dominant facts and forces at work at a given time.

Sensor and intelligence correlation types fall into three general categories: target-to-target, target-to-track, and track-to-track (Exhibit 6). The type of correlation used in a given situation depends heavily upon the type and currency of the data in the current correlated data base, the type and latency of the data to be correlated into that data base, and the needs of the system.

4.2.1 TARGET-TO-TARGET

Target-to-target correlation is most effectively used in a target-rich environment with continuous coverage by sensors. This correlation type generally uses either a statistical or a heuristic method to associate a report with an element in the data base containing only a current state descriptor. This algorithm is well utilized in situations where there is little or no latency in the data.

4.2.2 TARGET-TO-TRACK

In the situation where one of the items to be correlated is a track (a track represents the accumulated information concerning a target) and the other is a state descriptor, target-to-track correlation is used. The track may be derived from a data base that includes historical information, or a track generating sensor source. The algorithms used must account for the fact that there may be no node in the track near the point of correlation. This algorithm is best used for posterior updates to tracks through the use of data with a high degree of latency.

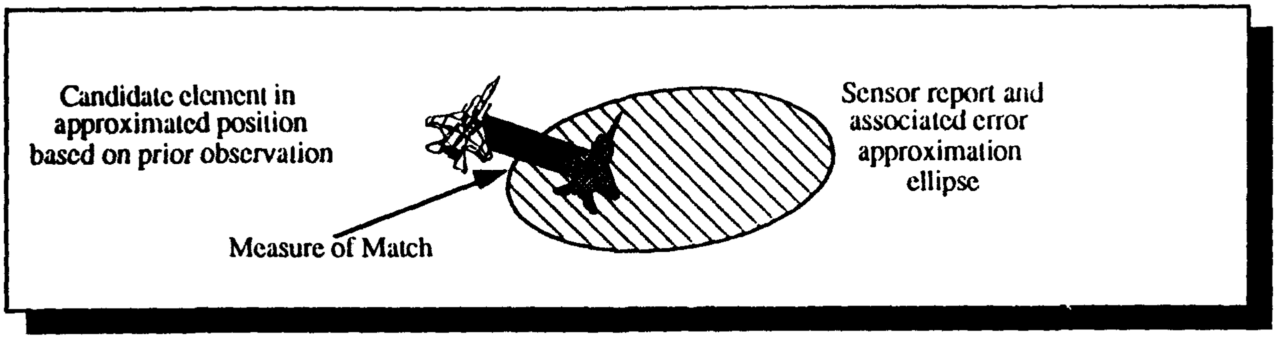
4.2.3 TRACK-TO-TRACK

Track-to-track correlation is used in situations where tracks are generated by two or more different sources and these tracks are to be combined to form a single track. In this case the problem is approaching the level of fusion rather than simple correlation. Many intelligence sources generate and feed tracks rather than individual reports. In addition, track-to-track correlation can be very valuable in cases where the update rate of the different sensors involved is significantly different. Target-to-target or target-to-track correlation may be applied to each source individually, and then track-to-track applied after that.

4.3 CORRELATION ALGORITHMS

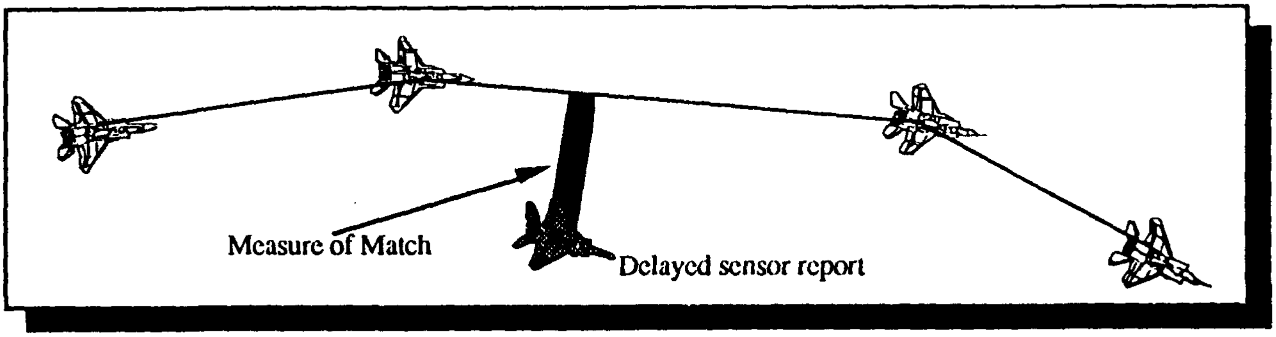
The algorithms used for sensor correlation can be divided into two main categories: Statistical Evaluation and Artificial Intelligence.

Exhibit 6
Basic Correlation Types
(Two-dimensional for example purposes only)


Candidate element in approximated position based on prior observation
Sensor report and associated error approximation ellipse
Measure of Match

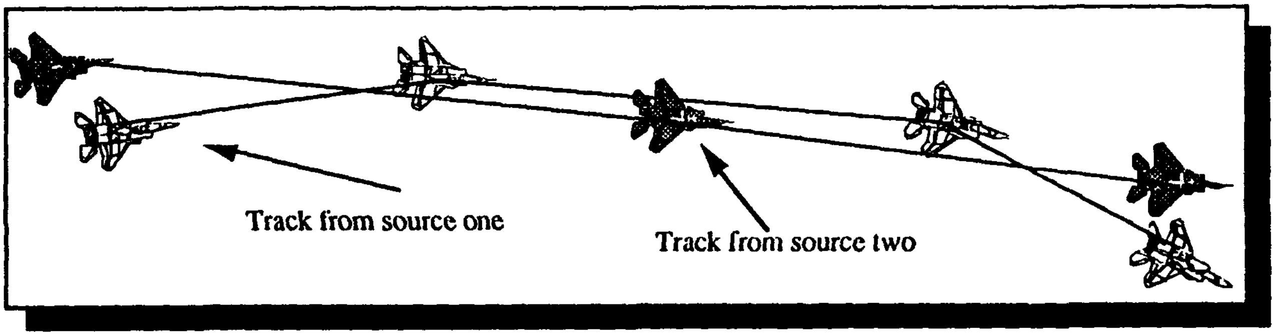
Target-to-Target

This is the most straightforward correlation type. Target-to-target correlation uses either a statistical or a heuristic method to associate a report with a unitary stored element which has been updated based on prior observations to match time stamps. This type of correlation generally works best in an environment where a high density of reports is available.



Target-to-Track

The target report is associated against a track which represents the accumulated information concerning an element (from a data base or track generating sensor source). In this case no effort is made to approximate position so as to match the time stamp of the sensor report. One case where this type of correlation is ideal is in the case of highly time-skewed target reports.



Track-to-Track

Track-to-track correlation is used in situations where tracks are generated by other assets and these tracks are to be combined. Many sources generate and feed tracks rather than individual target (sensor) reports. In addition track-to-track correlation can be very valuable in cases where the update rate of the different sensors involved is significantly different.

4.3.1 STATISTICAL METHODS

The collection of statistical methods presented below represent algorithms from a very small, selected subset of the correlation, pattern analysis, tracking, and probability models relevant to intelligence correlation and NCTI problem areas.

Bayes' Rule is a classic statistical approach to correlation based on pattern classification and a priori knowledge. This method provides a useful approach to problems where predicted behavior of targets to be correlated is available. The Army's Duplex Army Radio/Radar Targeting (DART), and the Enemy Intention Assessment Aid (INT) use this technique extensively.

Vague Probabilities were formulated to address the premise that basic theories of probability do not realistically handle the vagueness of sensors and targets as found in real life situations. The theory proposes that ranges of probabilities, say of a target and a stored element correlating, are a much more realistic representation of correlation than a simple measure. Vague probabilities are most often used in multihypothesis correlation systems, where several possible correlations are stored and reported.

The Kalman Filter is one of the grandfathers of statistics upon which correlation algorithms may be based. The Kalman Filter is a recursive algorithm to predict the state of an entity at time *n+1*, given knowledge accrued on it to time *n*. It is most usefully employed in target-poor environments for which sensors provide very high update rates.

Other techniques include Optimal Estimation Technique, the Nearest Neighbor Rule, the Mahalanobis Distance, and a host of others.

4.3.2 HEURISTIC TECHN!QUES

There are situations in which the nature and density of the elements to be correlated preclude the use of statistical methods. In these cases a heuristic approach is used. Heuristic algorithms are well suited to situations in which incomplete data produces a state in which the number of degrees of freedom for statistical analysis makes a statistically significant determination impossible. A selection of heuristic techniques which are applicable for correlation of intelligence information includes the following.

The Generate and Test Paradigm is an AI technique often used to choose between the "finalists" found by other methods. It is based on comparing calculated "ideals" against observed values. It is quite useful in some identification operations as well as for correlation. Generate and test solutions are excellent discriminators, but are highly compute intensive.

The General Problem Solver (GPS) paradigm utilizes a pseudo-state transition machine, the components of which are the current state description, a goal state description, and procedures designed to reduce the difference between the current state and the goal state. It is frequently used in algorithmically tasking sensors to resolve ambiguity arising from standard correlation.

A number of additional techniques have shown applicability as well. The Procedural Representation of Knowledge can be used as a symbolic correlation technique and is well suited to some of the more nebulous intelligence sources; and AND/OR trees are a specialized form of the data tree ALPHA-BETA decision making technique which can perform correlation based on game playing strategies.

Even simple Production Systems, based on "IF...THEN..." structures, can be used to solve problems in the "fuzzy" domains. They are particularly useful in hybrid correlation systems to pick between competing algorithms for a given situation. Production Systems have been implemented in the Advanced Information Presentation System, the Dynamic Air Order-of-Battle Aggregation Aid, and the Threat Evaluation and Countermeasure Agent.

Action-Centered and Object-Centered Control are two additional control techniques for determining which procedures will perform which actions in a system. In this technique, a control process "knows" which subprocesses or groups of subprocesses exist to do certain tasks. It also knows at any given time which subprocesses are free, and the relative merits of picking one process or group over the others to do a job.

4.4 NCTI SENSITIVITY ANALYSIS

Each of the parameters used in an actual NCTI scenario is subject to a number of error modes and error rates. It is important to determine in which parameters error causes gross problems, and in which a certain degree of error is permissible. This type of sensitivity analysis is one of the most crucial activities that can be performed by NCTI researchers.

This analysis must address the problems of data latency and tracking misregistration from different sensor sources. Other analyses concerning correlation erro , dissemination delays, and the integration of status data elements (such as AOB) can also be performed. All of this can be performed in the context of an analysis which considers national, theater, and tactical intelligence sensors and dissemination systems.

Sensitivity analysis, especially against a large number of relatively independent variables are present in an NCTI problem space, can be a very statistically and computationally demanding exercise. The completion of such a task requires the use of automated tools which can generate a sensitivity state-space or envelope for a number of parameters.

Analysis of NCTI parameters might include, but not be limited to, sensor availability, sensor accuracy, sensor timeliness, and track generation error modes. Similarly an analysis of the correlation algorithms can focus on the accuracy of the fused data and tracks derived, and the tactical utility of the data as disseminated to combatants.

These analyses should be made with respect to their effect on the outcome of air engagements, specifically the resulting probability of kill (Pk), reduction of fraternal and collateral damage, and friendly asset survivability factors. Parameters included should be those which relate to the following.

4.4.1 CONFIDENCE LEVELS

Target aircraft ID confidence levels are associated with existing identification processes prior to distribution. This probability of identification (Pi) is defined as the probability of an identification declaration being issued (expressed as a value in the range 0 to 1, inclusive) times the probability that the identification is correct (as assigned by the entity making the declaration and also in the range 0 to 1, inclusive). This analysis technique allows a quantitative resultant probability of kill (Pk), probability of fratricide (Pf), and probability of collateral (civilian) damage (Pc) to be determined as the result of the observed range of confidence levels for individual sensor types.

4.4.2 ID LEVELS

The identification of airframes is hierarchical. At the physical level an identification of the airframe gross type, specific airframe identity, and stores/control surface position data may be determined with finer resolution data. As data resolution increases the limits of physical attribute data, aircraft weapons load, drop tanks affixed, JEM, etc. may be determined.

4.4.3 EFFECTS OF TEMPORAL DISTORTION

By determining the sensitivity of the engagement model to varied temporal distortion of disseminated information, a quantitative analysis of the effects of temporal distortion (transmission delays) in the NCTI / HTI arena can be provided.

4.4.4 IMPACT ANALYSIS FOR WEAPONS DEPLOYMENT

To compute the weapons deployment impact it is necessary to determine the sensitivity of the weapons deployment portion of the engagement to varied temporal distortion of disseminated information. Within the context of the temporal analysis (i.e., the combat model selected will be supplied with the same tactical data with higher and higher degrees of temporal displacement), an accurate statistical model of BVR weapons such as AMRAAM, will give definitive thresholds of acceptable latency for tactical weapons deployment based on offboard data during critical mission phases.

4.4.5 TRACKING ERROR ANALYSIS

Tracking error analysis involves analyzing the effect of geolocation tracking errors from multiple intelligence sensors upon fusion and correlation into a simple track and ID report. These simple reports, which will be standardized in terms of geolocation expression and units, will be subsequently fused with an advanced multisensor fusion algorithm associated with the C3I/TACC architectures.

5.0 BACKPLANE UTILIZATION

The use of the NCTI Sim Mod I Backplane to perform model integration tasks must follow accepted design principles, and use established implementation strategies. Implementers should have expertise in C, Unix, NCTI , and sensor correlation areas.

The NCTI Sim Mod I Backplane allows all software implementation to be performed using the programming language upon which the original model is based, and using the C programming language in which the Mod I Backplane is implemented.

5.1 USE OF MODEL INTEGRATION LIBRARY RESIDENT IN MOD I

The integration of models into the NCTI Sim Mod I Backplane environment was performed based on the design developed through the considerations discussed in the previous section. Having developed detailed data translation maps and identified the points of change / interface within the target sensor model, the actual implementation involved three basic steps. The first step was the conversion of the local data so that it is compatible in form and content to the standardized Backplane form. The second step was the construction of mechanisms to allow the communication of data and timing information to and from the Backplane. The final step was informing the Backplane that the model is available, where it is located, and what it is good for, so that the model could be integrated into future experiments.

The NCTI Sim Mod I Backplane architecture includes a large library of data translation functions, communications functions and templates, and convenience routines to accomplish this task. Through the use of this library it was possible to incrementally implement a model (i.e., implement the attachment points in stages). This form of rapid prototyping allows a model to be used in limited scenarios for validation prior to full scale integration.

5.2 DATA REQUIREMENTS AND UNITS CONVERSION

The target and sensor information which forms the basis of the Mod I Backplane system is based on specific units of measurement and data forms. The sensor models and their associated processing components, however, will almost never be in the same units.

Target reports may be expressed in a number of ways including spherical representation (i.e., range, bearing, and azimuth to a target from a fixed or mobile point in 3-D space), or in geographic coordinates in terms of latitude and longitude within one of the several standard map projections, or northings and eastings in the UTM projection.

The Mod I Backplane includes libraries of functions to handle almost all of the expected units of measurement (e.g., frequency, distance, time, power, heading, etc.) and allows them to be converted into standard units. This set of routines provides a large number of unit conversion operations. The number of functions is large, but some examples include conversions so that frequency will be expressed in Mhz; speed in meters/sec; heading in radians on the scale from 0 to 2π, with 0 representing true north; altitude in meters; etc. Furthermore data templates allow the arrangement of data elements to be set to conform with a standard target report and communication style to be designed as the result of the analysis performed during the model acquisition.

5.2.1 SPATIAL CONVERSION ROUTINES

The NCTI Sim Mod I Backplane represents spatial data using Arc-second Raster Coordinate (ARC) representation. This must be done so as to be compatible with the Common Mapping System (CMS). CMS is the Air Force's recently adopted standard system for the preprocessing, storage, and utilization of cartographic and imagery data. It forms the baseline for the AFGIS, ULPI, and MSS II programs, and is the foundation for RAAP and most current and planned Air Force efforts that involve spatial data. Within this framework all distances are represented in meters.

Routines are included to convert from Lat-Long (in any projection), UTMs, and certain polar representations to the ARC form.

5.2.2 KINEMATICS

Airframes within the ground and perceived truth data bases are described using a set of kinematic values. These include:

√ Heading The X, Y, and Z deflections describing the axis of travel. Values are in degrees relative to due north where the X axis is E-W, the Y axis is N-S, and Z reflects elevation. (NOTE: For directional emitters associated with airframes, the Look portion of the emitter description is NOT relative to this heading; it is absolute.)

√ Velocity The current velocity of the airframe along its heading in meters per second. Stationary airframes with velocities of 0 are allowed.

√ Engine_status Percent, from 0.0 to 1.0, of maximum engine output currently being applied. Engine status values in excess of 1.0 (100%) are allowed and represent afterburner, where allowed.

5.2.3 RADIO FREQUENCY (RF) EMISSIONS

NCTI processing is very involved with the processing of RF information. Unfortunately RF systems can be described numerically in a large number of ways. The Mod I Backplane uses the most common descriptive set which includes:

√ Frequency The center frequency of the emitter in Hz.

√ Pulse_repetition_interval The PRI / PRF of the emitter in Hz.

√ Pulse_activity See the appendices of the NCTI Sim Mod I Backplane User's document for a complete list.

√ Pulse_duration Pulse duration of the emitter in milliseconds.

√ Scan_type See the appendices of the NCTI Sim Mod I Backplane User's document for a complete list.

√ Scan_rate The scan rate of the emitter in Hz.

√ Antenna_polarization The polarization of the antenna in degrees.

√ Bandwidth The bandwidth rate of the emitter in Hz.

√ Location X, Y, and Z coordinates of the emitter with respect to the centroid of the model space in meters.

√ Look — X, Y, and Z deflections describing the axis of emission for a directional emission source. This has no meaning for omnidirectional emitters. Values are in degrees relative to due north where the X axis is E-W, the Y axis is N-S, and Z reflects elevation.

√ Divergence — The amount (in meters) that a directional beam signal spreads away from an ideal cylinder per 100 meters of travel. (NOTE: More complex functions of divergence and emission patterns must be supported externally; this data for approximation only.)

√ Power — The emissive force in watts at the antenna or feed horn.

√ Dissipation — The mean percent reduction (expressed as a number from 0.0 to 1.0) in the power of the signal per 100 meters. (NOTE: More complex functions of dissipation and emission patterns must be supported externally; this data for approximation only.)

The Mod I libraries contain limited functions to convert data to these units. Normally any required conversions are straightforward and can be derived from the CRC™ or other standard reference guide.

5.3 COMMUNICATION WITH BACKPLANE

Communication with the NCTI Sim Mod I Backplane is accomplished using Unix socket based processing. This type of communication supports a large number of different programming languages, computer architectures, and operating systems.

All of the inter-model data communication tasks which occur within the Mod I environment are based on the transmission of a standardized data message packet by a model to the Backplane, and on a reply generated. This highly structured step-in-step type of communication assures that the Backplane arbiter stays firmly in control of the overall simulation at all times.

5.4 MODEL STARTUP, INITIALIZATION, SYNCHRONIZATION

The Backplane is equipped with a very powerful system to handle the configuration and startup of the multiple models and processes that make up an NCTI Sim Mod I run. Subject to the constraints of logical information flow, there is no limit to the channels used by the models in communicating with each other. However, the Backplane must be initialized somehow to let it know what simulation models will be executing and what sort of interdependencies will exist between them. Under the NCTI Sim Mod I effort a process to handle this was developed.

Once a model has been identified and is ready to be integrated into the Model Management System, the software representing that model must be placed into the right location. The executable version of the model must be placed into the directory corresponding to the type of model. The following table illustrates this.

Type of Model	Directory Location
Sensor	/home/ncti/backbone/collection
Fusion	/home/ncti/backbone/fusion
Dissemination	/home/ncti/backbone/dissemination
Engagement	/home/ncti/backbone/engagement

After the model has been placed in the correct directory, the Model Management System will automatically know where to look for specific models that the operator would like to use for a particular run of the system.

The operator may build a system like the one described above from scratch or may choose from a list of previously built configuration files. To choose from a list, the user clicks on the SELECT CONFIGURATION button and is given a choice as to which configuration he desires. To build from scratch, the user selects any of the buttons, ADD SENSOR MODEL, ADD FUSION MODEL, ADD DISSEMINATION MODEL, SELECT ENGAGEMENT MODEL, ADD SENSOR QUEUE, and ADD DATABASE, to build a system one element at a time. Once all the elements have been built, one must determine the dependencies between the elements. For example, to tell the system that data from Sensor1 must go to Queue1, first click on the SET DEPENDENCIES button followed by the Sensor1 button and then the Queue1 button. A large arrowhead will then be drawn from Sensor1 to Queue1. Also, if the user selects a dependency that really does not make sense, a message will be displayed saying that the operation to be performed is not correct. The Model Management System has quite a bit of error checking in this regard. Once a configuration has been decided upon, the user may then save it for later use by selecting the SAVE CONFIGURATION button.

Selecting the EXIT PROGRAM button will terminate the Model Management Program and generate an input file to the Backplane process. The Backplane then uses the information in the file to launch all of the required simulation models automatically in addition to the man-machine interface.

5.5 INTEGRATION OF A MODEL

Integration of models involves the design of changes in functionality so that the original model elements can be used in the NCTI Sim Mod I environment. This analysis and design will result in an identification of the locations within the original model source code that require alterations as well as a description of those changes.

Changes to existing models generally involve:

√ Getting data to the model so that the model operates against a data set which is representative of the ground truth data found in the system.

√ Getting results from the model.

√ Controlling and synchronizing the model with the rest of the system.

It is reasonable to implement these changes in a cursory fashion to evaluate the desirability of a model prior to the full scale integration of all hooks. This strategy of rapid prototyping can help eliminate or validate models in a cost effective manner.

The remainder of this section details some of the design and analysis considerations that must be addressed in each of these three areas.

5.5.1 INPUT DATA REQUIREMENTS ANALYSIS AND DESIGN

The first step in the analysis and design phase is to examine the data elements that the selected model requires for operation. These data items typically fall into the areas of sensor internal, model internal, and external.

Sensor internal data is that data which the sensor being modeled uses for its operation. It represents the information that the actual live sensor system would use in its operation. Items such as templates, timing and control parameters, and other sensor specific information fall into this category.

Model internal data is that information which is required by a sensor model, but which would not be required by the actual sensor. An example of this type of information is data used to calculate a signal return from a particular airframe. In an operating sensor this information would be collected from the sensing elements themselves. Model internal data is, therefore, the data required to actually do the modeling.

Model internal data can be further divided into data which models the sensor system, data which represents the environment, and data which represents the thing being sensed. Of these three types, data which represents the thing being sensed is the data that will be most severely impacted by integrating the sensor into the Mod I Backplane.

External data is all else which does not fall into either of the above categories.

5.5.1.1 Data Contents

The first step is to perform an analysis of the contents of the data used by the sensor model. It will be necessary to build a list of all of the individual data elements stored by the sensor. Then it will be necessary to build a map which indicates, for each data element, if and where in the Backplane data structure the corresponding data may be found.

5.5.1.2 Data Representation

An examination of the way in which data is represented within the model is the next step in the design process. In order to design an interface into the existing data the following information must be determined:

Memory Model — The type of memory model used by the implementers of the sensor model must be determined. Items which must be considered are if the parameters are stored in memory, or in a data base; if the variables are passed by reference or by value; and if the variables are in simple tables, linked lists, structures, or some other format.

Data Type — Once the memory model has been examined to determine the methodology needed to access the data, it must be determined what the data type is—numerical, Boolean, textual, or other, or an integer or real number, and the type of representation—least-to-most, modulo-8, exponential, or packed decimal.

Units of Measure — Once enough information has been gained to allow the data elements to be accessed and put into a legible form, it must be determined if the units of measure are meters, yards, feet, or miles, and if the angles are in degrees or radians, and what the starting point is.

Dependency — Finally it must be determined what sorts of dependencies exist. In some cases data items are merely pointers into local tables. In other cases changing one variable will have a considerable effect on another.

5.5.1.3 Data Access

After analyzing the data elements that are provided to the sensor model by the Backplane, the next step is to determine how this could best be done. The two most straightforward approaches are local data mirroring or "snapshotting", and local data intercept. To a large extent the internal construction and behavior of the model in question determined which of these is used. Exhibit 7 illustrates these two techniques.

5.5.1.3.1 Local data mirroring. Local data mirroring, also called "snapshotting," involves replacing the entire internal data structure of the sensor model with data retrieved from the Backplane data base once each simulation time period. The result is that the two data bases agree at the start of each sensor model run and the results of the run will be a series of target reports which reflect a data set and environment which is very nearly identical to the current ground truth.

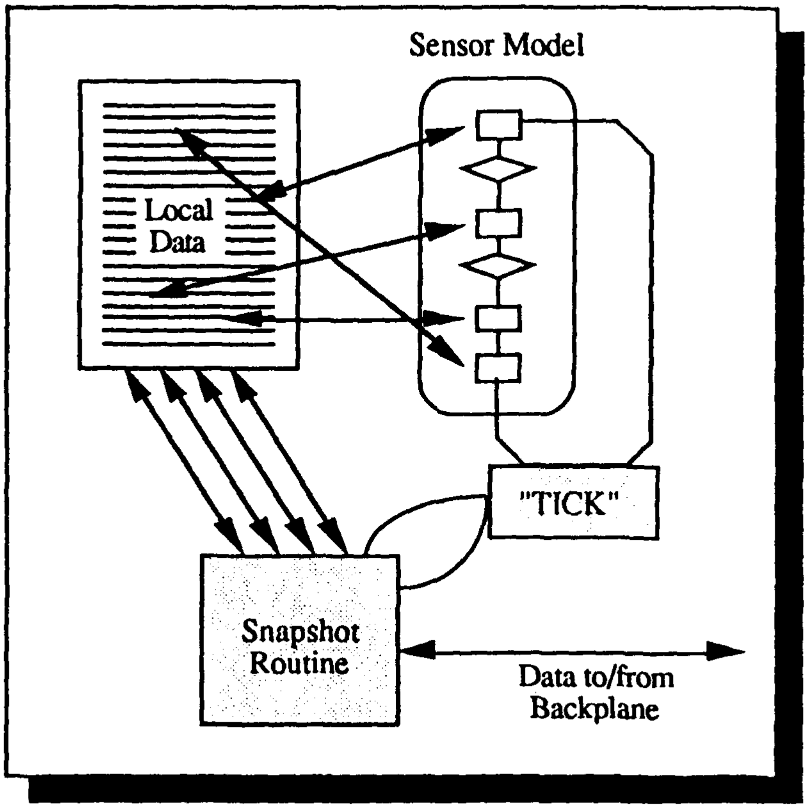
Note that this representation will be very close, but *not* identical to the current ground truth. Under the local data mirroring scheme, updates that occur after the start of the simulation time step, but prior to the completion of execution of the sensor model, are not reflected until the next time step. While this is less than perfect it is very accurate if the duration of the simulation time step is selected with care.

This technique is the technique of choice when integrating sensor models which rely heavily on data tables that are not accessed through a set of common interface routines, but instead are accessed directly at a large number of points throughout the model code. In cases where the local data is stored in a way that mandates mirroring, a snapshot routine was designed to be executed at the posterior end of the tick synchronization cycle.

5.5.1.3.2 Local data intercept. Certain sensor models are implemented using common data access routines. In some cases these routines access commercial (or custom) data base managers; in others they are just standardized techniques for data access adopted by the implementers.

In either case, these common routines may be modified to access the Backplane's data sources rather than the local data. Depending on the interdependencies between data items these common routines are either supplanted entirely, or run in conjunction. In cases where local data access routines are used in a consistent fashion the implementation of intercept routines are included in the design.

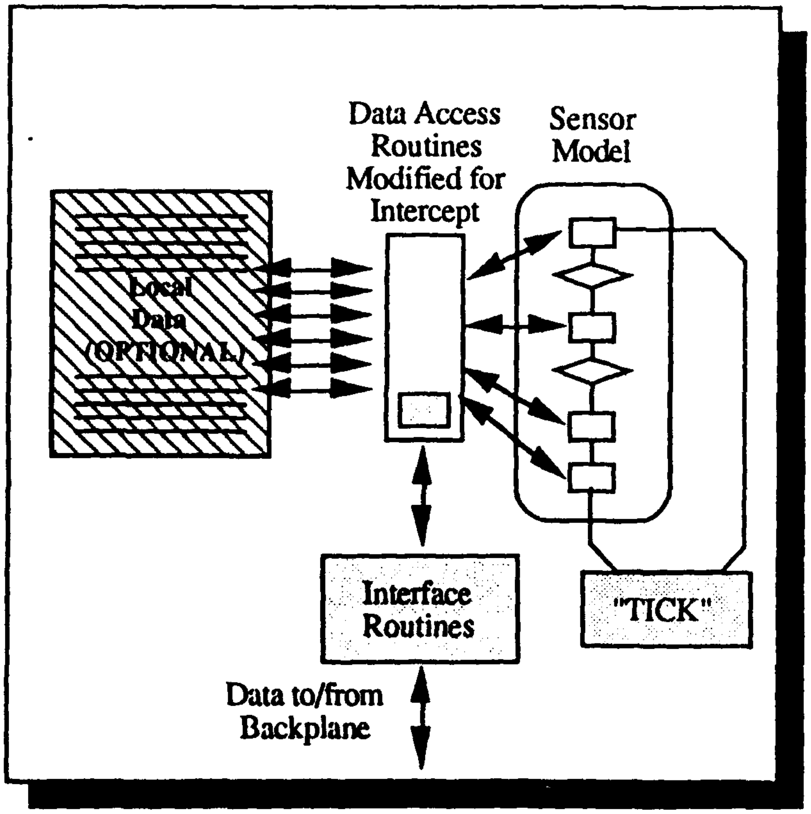
Exhibit 7
Techniques for Sensor Data Interfacing



In Local Data Mirroring the entire internal data structure of the sensor model is updated with data retrieved from the Backplane data base once each simulation time period.

At the end of the period the Backplane is updated with the changes made to the local data (if any).

The result is that the two data bases agree at the start and end of each sensor model run.



In Local Data Intercept the routines that the model uses to access data are modified to get the data from the Backplane.

The routines can also update local data if that is necessary, or the local data base may be deleted.

The result is that local sensor data and ground truth data always agree throughout sensor model run.

5.5.2 OUTPUT ANALYSIS AND DESIGN

The expected output of a sensor model is one of the more rigidly structured portions of the NCTI Sim Mod I Backplane. This rigidity stems from a number of sources but the two predominant ones are the necessity to support a comprehensive ground/perceived truth data base set, and, more importantly, the need to feed a wide variety of correlation and fusion algorithms.

The design and implementation of the Mod I Backplane was based on a comprehensive analysis of sensors, sensor models, and correlation and fusion algorithms. The decision was made that the standardized target report output format for integrated sensor models would be above the engineering level; that it would include identification information (but that such information need not be supplied by the sensor directly); and that reports of electronic emissions and airframe parameters would be the ones supported by the in-band channels.

5.5.2.1 Sensor Model Analysis and Design

The process of integrating the selected sensor model(s) into this rigid structure was begun in a fashion similar to the analysis previously described. The data output from the sensor model was examined and a mapping between the data thus derived and the data elements expected by the NCTI Sim Mod I target report message was generated. Exhibit 8 illustrates the contents of the standard Mod I airframe target report.

5.5.2.2 Report Dissemination Strategies

In designing the interface to the model it is important that the time density of the output data be representative of the density that the *actual sensor system* produces. Since the Mod I Backplane system allows statistics to be collected for each of the modeled communications pathways, the amount of information disseminated by the sensor model should be realistic. In performing the design we used either a series of null ticks (i.e., in the case where the data rate from the model exceeds the actual sensor), or a strategy of double (or more) transmissions of the same information (i.e., in cases where the actual sensor data rate exceeded the model's).

5.5.3 TIMING ANALYSIS AND DESIGN

In the analysis discussion the notion of model execution timing was briefly mentioned. Perhaps one of the most difficult tasks involved in integrating a number of different models and algorithms from diverse sources is getting them all to run in a synchronized fashion. The Mod I Backplane uses a set of semaphores known as "ticks" to synchronize all of the client models resident on the Backplane during a run.

Ticks come in major and minor forms and are also sometimes present in both normal and fast. It is incumbent upon the model integrator to utilize the tick semaphore structure in the proper fashion. The Backplane has no ability to throttle the client models in any other meaningful way (i.e., resource control semaphores are not time sensitive).

5.5.3.1 Minor vs. Major Ticks

Both minor and major ticks occur at the normal rate. The only difference is that minor ticks are issued when no targets are proximal enough to the sensor in question to be considered critical.

Exhibit 8
Template for Mod I Function

Send Target Report Airframe Library Function

Queues a target report concerning an airframe for transmission to the correlation / fusion / control element

Platform_status Integer
Indicates the status of this platform as follows:
0 Dead, 1 Active, 2 Lame, 3 Indeterminate

Platform_allegiance Integer
Indicates the force allegiance of this platform as follows:
1 Blue force combatant or platform under direct blue force control.
2 Red force combatant or platform under direct red command and control.
3 Green, Blue force ally but not directly under the control of a blue control station.
4 Pink, Red force ally of unknown command and control characteristics.
5 Gray platform, not a combatant.
6 Unknown.
(NOTE: This information is available from sensors which provide NCTI information and should otherwise be 0.)

Location Real(3)
X, Y, and Z coordinates of the platform with respect to the centroid of the model space in meters.

Heading Real(3)
X, Y, and Z deflections describing the axis of travel. Values are in degrees relative to due north (where the X axis is E-W, the Y axis is N-S, and Z reflects elevation). (NOTE: For directional emitters associated with platforms the Look portion of the emitter description is NOT relative to this heading; it is absolute.)

Velocity Real
The current velocity of the platform along its heading in meters per second. Stationary platforms with velocities of 0 are allowed.

Engine_status Real
Percent (from 0.0 to 1.0) of maximum engine output currently being applied. Engine status values in excess of 1.0 (100%) are allowed and represent afterburner (where allowed). Negative value indicates unknown.

Stores Integer(16)
The stores on board the airframe (if known). See the appendices of the User's Guide document for a list of stores.

This minor / major flavor of ticks may be used by hierarchical models to switch between the low and high resolution internal calculations.

5.5.3.2 Normal vs. Fast Ticks

Under almost all operating conditions all of the models will receive normal ticks. These ticks occur once per simulation time period and the standard Backplane WAIT function will suspend the operation of the sensor (or any other) model, release all resources, and hibernate the process until the next normal tick occurs.

When a weapon is in flight (known as a weaponeering event), there will be a series of fast ticks issued to any process which has indicated the ability to accept them. Normally this includes only the combat model since only onboard avionics can detect and relay information concerning weaponeering fast enough to be useful. It is perfectly acceptable, however, for sensor models to accept fast ticks.

5.5.3.3 Event-Driven and Monte Carlo Models

When designing the interface between the tick handler and the sensor model it is important to distinguish between event-driven and Monte Carlo style sensor models, and deterministic sensor models.

While deterministic sensor models are the rule, there are a number of examples of event-driven models. In an event-driven model the internal architecture of the model is a finite state machine which is driven from state to state by external events. These events can be changed in state of targets, emissions from radio and RADAR sources, etc.

The integration of event-driven models is generally more difficult in the Backplane architecture since there are no provisions for the Backplane to signal events to sensors. The architecture is a strictly client requester/server one in which information about entities is never volunteered. The reason for this stems from the incredibly large number of event types the Backplane would need to be able to signal in order to be useful for event-driven processing.

In instances where event-driven models are to be integrated, a local data representation of targets of interest is kept by the model integration routines. This model is then compared to the contents of the ground truth data base held by the Backplane on a periodic basis (i.e., once per tick is the norm). When a discrepancy is noted the local data is updated and the proper local event types are generated. When a change of state is forced the appropriate target report (if any) can then be sent by an enqueue call added to the state machine destination node.

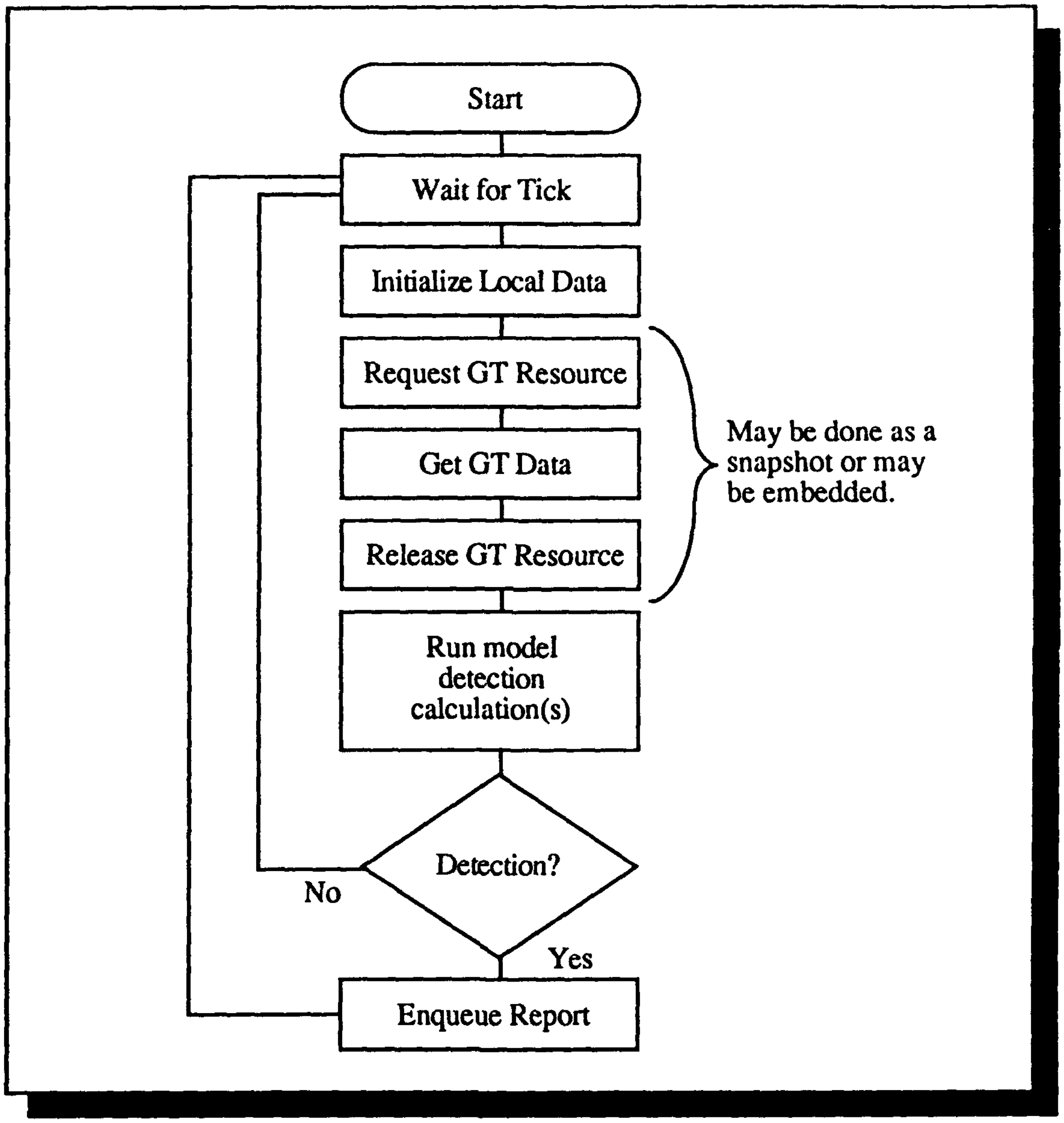
5.5.3.4 Deterministic Models

Deterministic sensor models are much more common than event-driven models. In a deterministic model, execution is begun, the model collects information, and performs calculations until a result is reached. Typically this result is either a detection or a failure of detection.

The integration of a model of this type is extremely simple, and is in fact the type of integration for which the Backplane was specifically designed. Exhibit 9 illustrates the integration of such a model into the Backplane environment.

As can be seen from this illustration the interfacing of a deterministic model requires that changes be made in three basic locations. First the flow control of the model is interrupted by a

Exhibit 9
Integration of a Sensor Model into the
Backplane Environment



call to the tick handler. (If the model does not currently run in a loop it is modified to do so.) Secondly, calls to the resource management and data access routines are embedded in the model code so that data is supplied from the Backplane data base. Finally a target report enqueue call is appended to the successful results branch of the model.

5.6 STRAW MAN MODEL CODE SAMPLE

The difficulty in describing the exact implementation strategy to be used in integrating a sensor model is evident from the previous sections. Listed below is a pseudo-code representation of an existing sensor model which has already been incorporated into the Mod I Backplane. This model (a RADAR model) was originally written in Fortran, and was hooked to the Backplane with C bridge functions.

The following code example illustrates the use of the integration library of the Mod I Backplane to attach an existing RADAR model to a scenario.

```
/****************************************************
** Sample routine to implement the driver portion of a
** gimbaled RADAR attached to the nose of an airframe
** which flies a four cornered cap.
****************************************************/

Integer   Airframe_ID;
Integer   Emitter_ID;
Integer   Collector_ID;
Integer   Time_elapsed;

Main
{
/* Function to set up RADAR parameters */
External Void Initialize_Radar();


/* Emitter description variables */
Integer   Emitter_tvpe;
Integer   Operating_mode;
Real      Frequency;
Real      Pulse_repetition_interval;
Integer   Pulse_activity;
Real      Pulse_duration;
Integer   Scan_type;
Real      Scan_rate;
Integer   Antenna_polarization;
Real      Bandwidth;
Real      Location[3];
Real      Look[3];
Real      Divergence;
Real      Power;
Real      Dissipation;



/* Variables to describe the collector platform */
/* NOTE: The actual flying of the platform will */
/* be performed by the stub function Mov_Platform */
/* which also takes care of re-aiming the emitter */

External Void Mov_Platform();

Integer   PCollector _ID[16];
Integer   Airframe_type;
Integer   Platform_status;
Integer   Platform_allegiance;
Real      PLocation[3];
Real      Heading[3];
Real      Velocity;
Real      Engine_status;
Real      Stores[16];
Integer   i;

/************************************************
Inform the Backplane manager that a sensor process is
being added to the environment.
************************************************/
Collector_ID = Declare_sensor('Test Collector',
     'Collector for testing purposes only');

/************************************************
Declare the emitter that makes up the RADAR
************************************************/
Initialize_Radar(     Emitter_Type,
     Operating_mode,
     Frequency,
     Pulse_repetition_interval,
     Pulse_activity,
        Pulse_duration,
        Scan_type,
        Scan_rate,
        Antenna_polarization,
        Bandwidth,
        Location,
        Look,
        Divergence,
        Power,
        Dissipation);

Emitter_ID = Declare_emitter('Test Radar,
        'Emitter_description field',
        Collector_ID,
        Emitter_Type,
        Operating_mode,
        Frequency,
        Pulse_repetition_interval,
        Pulse_activity,
        Pulse_duration,
        Scan_type,
        Scan_rate,
        Antenna_polarization,
        Bandwidth,
        Location,
        Look,
        Divergence,
        Power,
        Dissipation);

/*************************************************
Add the airframe that this Radar will ride on.
*************************************************/
For (i=0; i<16; i++)
{
        PCollector_ID[i]=0;
        Stores[i]=0;
}
PCollector_ID[0] = Collector_ID;
Airframe_type = 257;
Platform_status = 1;
```

```
Platform_allegiance=1;
Plocation[0] = -10000;
Plocation[1] = -32000;
Plocation[2] = 2350;
Velocity = 3477;
Heading[0] = 20000;
Heading[1] = -30000;
Heading[2]= 2350;

Declare_platform(PCollector _ID[16], Airframe_type,
    Platform_status, Platform_allegiance, Plocation,
    Heading, Velocity, Engine_status, Stores);


/*****************************************************
** Main Loop
*****************************************************/
While(True)
{
    Time_Elapsed = Synchronize_collector(Collector_ID);

    /*****************************************************
    !!!!!!!!!!Warning!!!!!!!!!!!
    ** Note: that Mov_Platform performs a Request_gt_resource !!!
    ** don't forget to release it !!!!!!
    *****************************************************/
    Mov_Platform(Time_elapsed, Plocation,Heading,
        Velocity, Engine_status);

    /*****************************************************
    *****************************************************
    Place the call to the original sensor code here.
    It will perform the DB retrieve and send the target reports
    then release the data base internally.
    *****************************************************
    *****************************************************/
    Old_Code();

    }
}
/* End */
```

MISSION
OF
ROME LABORATORY

Mission. The mission of Rome Laboratory is to advance the science and technologies of command, control, communications and intelligence and to transition them into systems to meet customer needs. To achieve this, Rome Lab:

a. Conducts vigorous research, development and test programs in all applicable technologies;

b. Transitions technology to current and future systems to improve operational capability, readiness, and supportability;

c. Provides a full range of technical support to Air Force Materiel Command product centers and other Air Force organizations;

d. Promotes transfer of technology to the private sector;

e. Maintains leading edge technological expertise in the areas of surveillance, communications, command and control, intelligence, reliability science, electro-magnetic technology, photonics, signal processing, and computational science.

The thrust areas of technical competence include: Surveillance, Communications, Command and Control, Intelligence, Signal Processing, Computer Science and Technology, Electromagnetic Technology, Photonics and Reliability Sciences.